Observability ohne Mühe

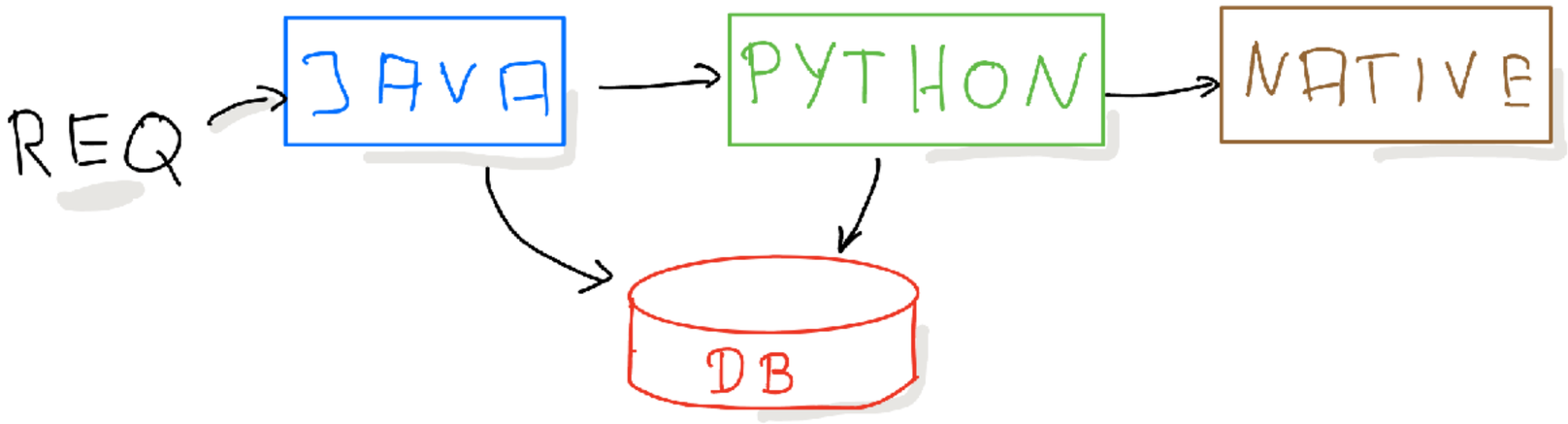
Oder wie die Einstiegshürde für Tracing ohne großen Aufwand überwunden werden kann.

Heiko Rupp, 9.7.2026, Java-Forum-Stuttgart

Red Hat

Szenario

Ein verteiltes System



Siehe auch https://www.pilhuhn.de/docs/distributed-tracing-quarkus/

Red Hat

Observability

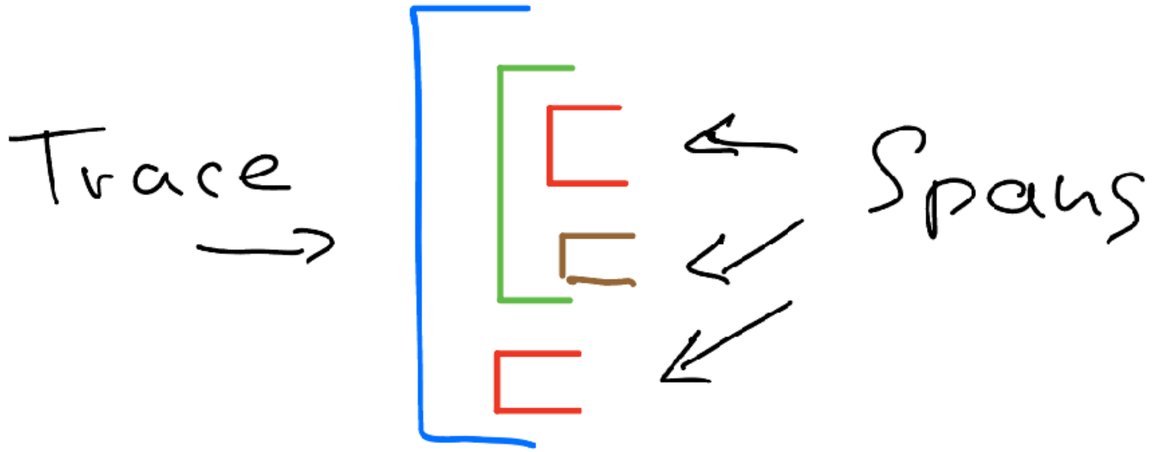
O11y wer?

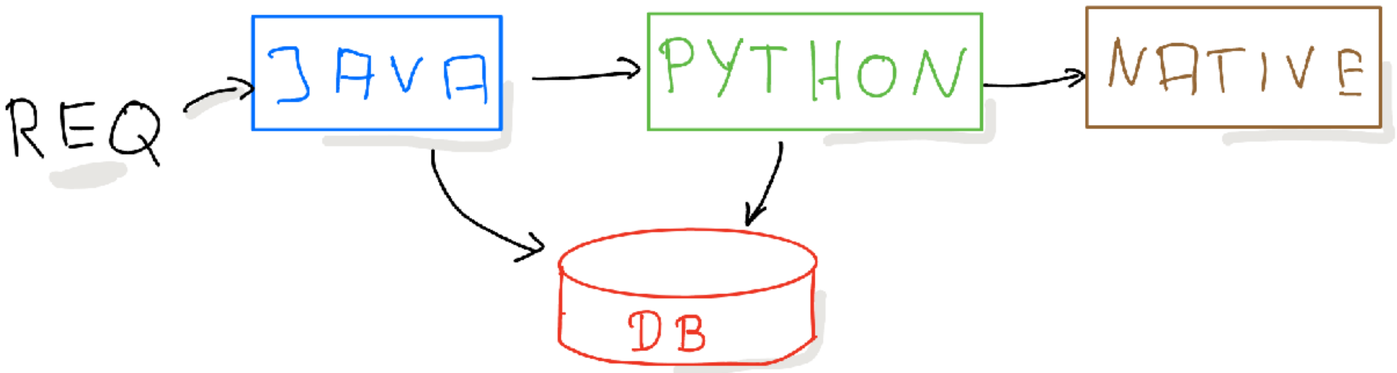
- Sammelbegriff für Metrics/Logs/Traces/…
- Anwendung als Blackbox
  - Kein Zugriff via Shell / Debugger / Profiler
- Reasoning via exportierter Signale
- Datenschatz
  - Performance- / Last-Vorhersagen
  - Canary-Deployments

Red Hat

Tracing: Theorie

Aufrufe nachverfolgen

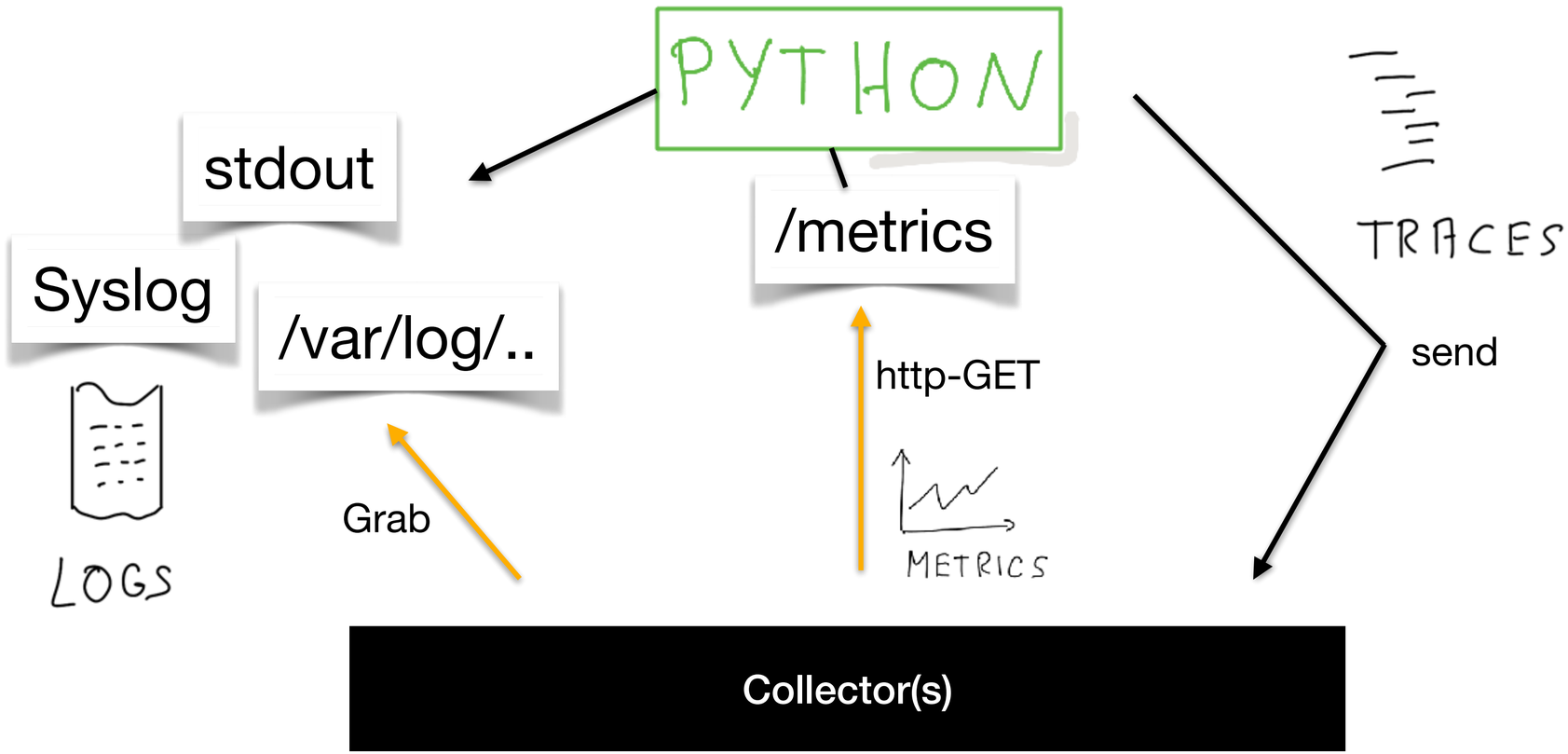




Red Hat

Wie komme ich an die Daten?

Wir müssen die Prozesse instrumentieren

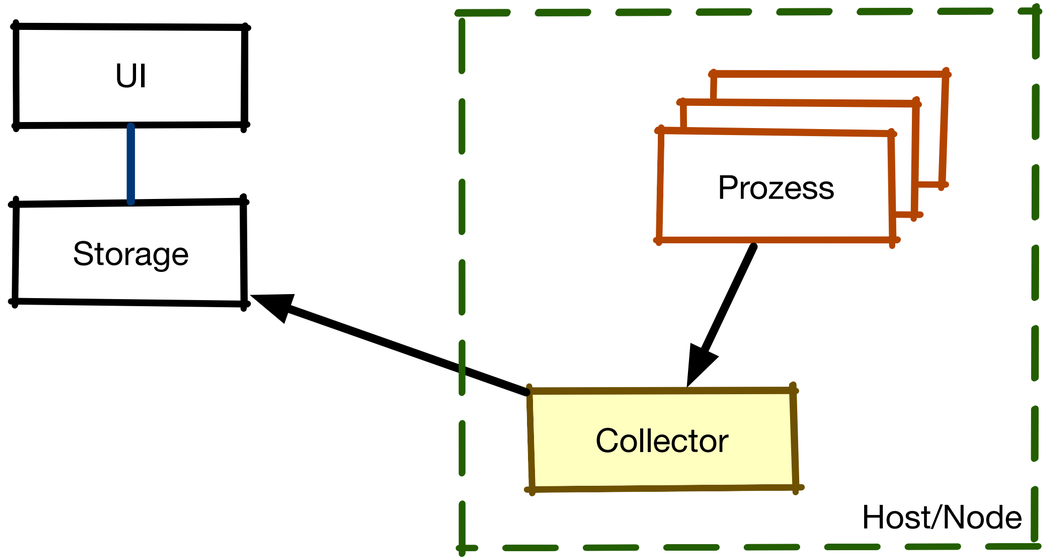


Red Hat

OpenTelemetry(.io)

Ein Ring, um sie alle zu binden

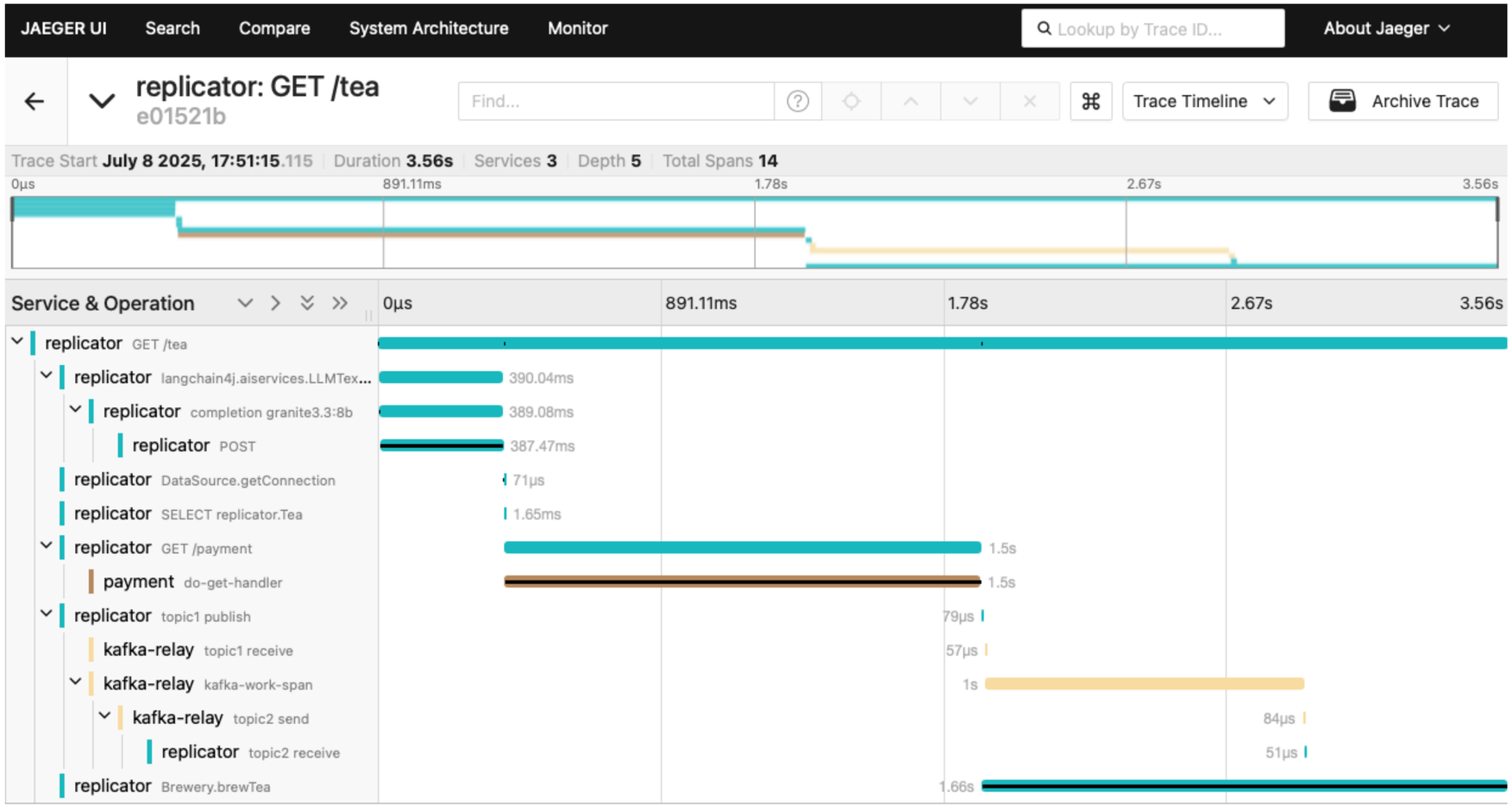
- Standard für API / SDK / Datenformat
- Grundsätzliche Konfiguration
  - service-name (`OTEL_SERVICE_NAME`)
  - Ort des Otel-Collectors
    - Standardport :4317/:4318




```
quarkus.application.name=replicator
quarkus.otel.exporter.otlp.endpoint=http://localhost:4317
```

Fokus auf Tracing

Beispiel eines Traces



Red Hat

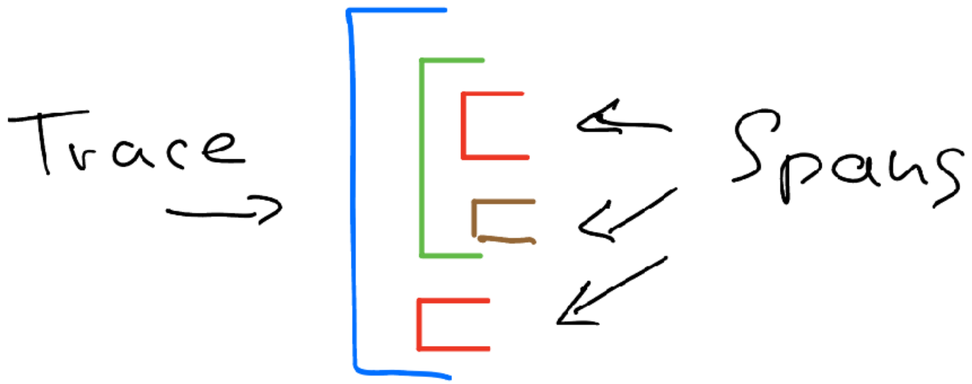
Wichtig: Kontext weitergeben

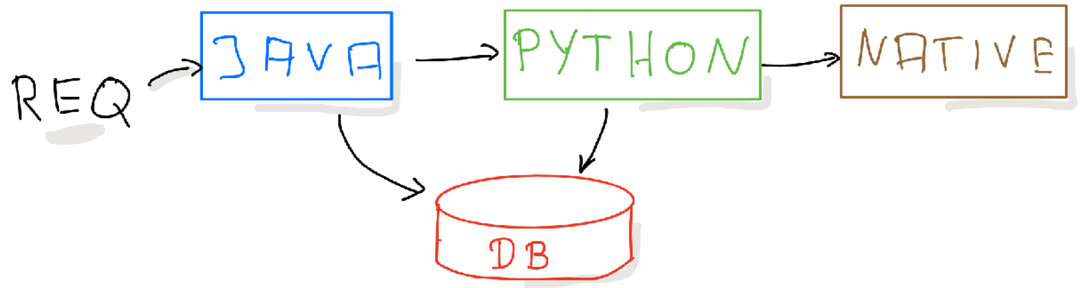
Chabos wissen, wer der Babo ist

HTTP-Header:

```
traceparent: 00-9312f0f2f06c32d85cfd1cfd5da5d1ee-85571b5267f9a042-01
```

Version TraceId SpanId Flags

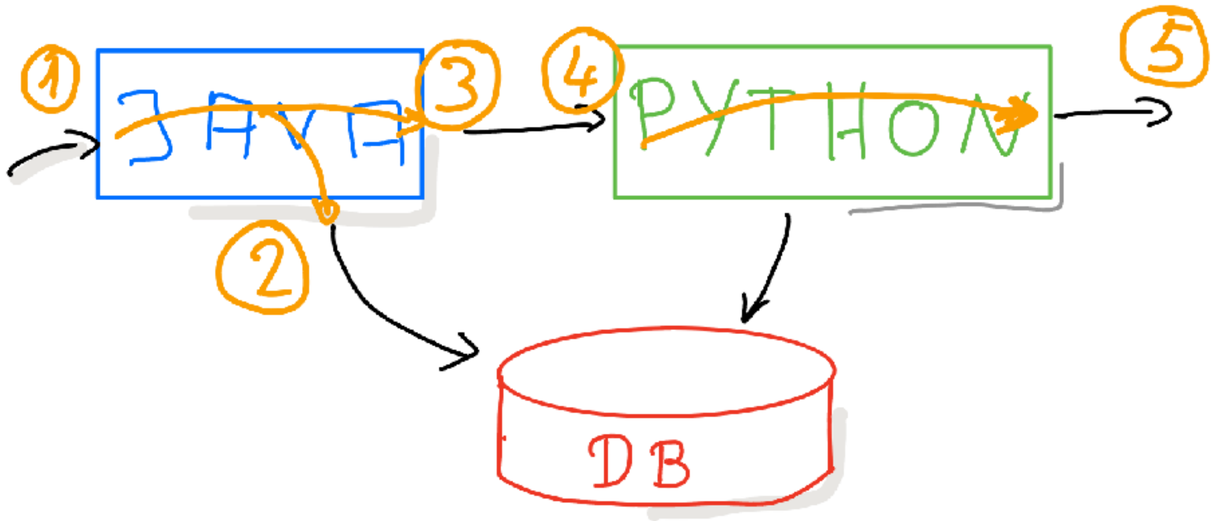




Red Hat

Wichtig: Kontext weitergeben (2)

Auch die inneren Werte zählen…



1->2, 1->3, 4->5 Kontext intern weitergeben

Red Hat

Optionen der Instrumentierung

Viele Wege führen nach Rom

- Nichts tun - keine echte Option
- Manuelle Instrumentierung
- Instrumentierungsbibliotheken
- (Java-)Agenten
- Sidecars / Proxies
- OBI

Red Hat

Manuelle Instrumentierung

Anstrengend aber detailliert

- Programmatisch
- Braucht Hilfe bei
  - Kontext-Propagation
  - Versand der Spans
- Einfaches Instrumentieren von Geschäftslogik
- Benötigt trotzdem SDK

```
with tracer
  .start_as_current_span(„get-handler",
  context=ctx) as span:
     span.set_attribute('req_line', req)
```

```
public boolean isPaid(String kind) {
    Span span = Span.current();
    span.setAttribute("p.kind", kind);
```

Instrumentierungsbibliotheken

Jemand anders hatte das Problem schon mal

- Vorhanden für populäre Bibliotheken

```
RequestsInstrumentor().instrument()
Psycopg2Instrumentor().instrument(
    enable_commenter=True,
    commenter_options={})
```

```
<dependency>
    <groupId>io.quarkus</groupId>
    <artifactId>quarkus-opentelemetry</artifactId>
</dependency>
<dependency>
    <groupId>io.opentelemetry.instrumentation</groupId>
    <artifactId>opentelemetry-jdbc</artifactId>
</dependency>
```

(Java-)Agenten

- -javagent

```
export JAVA_TOOL_OPTIONS=\
   "-javaagent:./opentelemetry-javaagent.jar"

export OTEL_SERVICE_NAME="java-with-agent"

java -m jdk.httpserver
```

Agenten

- LD_PRELOAD
  - Per Prozess
  - Oder `/etc/ld.so.preload`
- Kombination via Injector-Projekt

```
export LD_PRELOAD=/home/hwr/jfs2026/libotelinject_amd64.so

export OTEL_SERVICE_NAME=py-with-ld

export OTEL_EXPORTER_OTLP_ENDPOINT=http://localhost:4317

python -m http.server
```

Agent-/Library - Injection

Eine andere Art der Instrumentierung

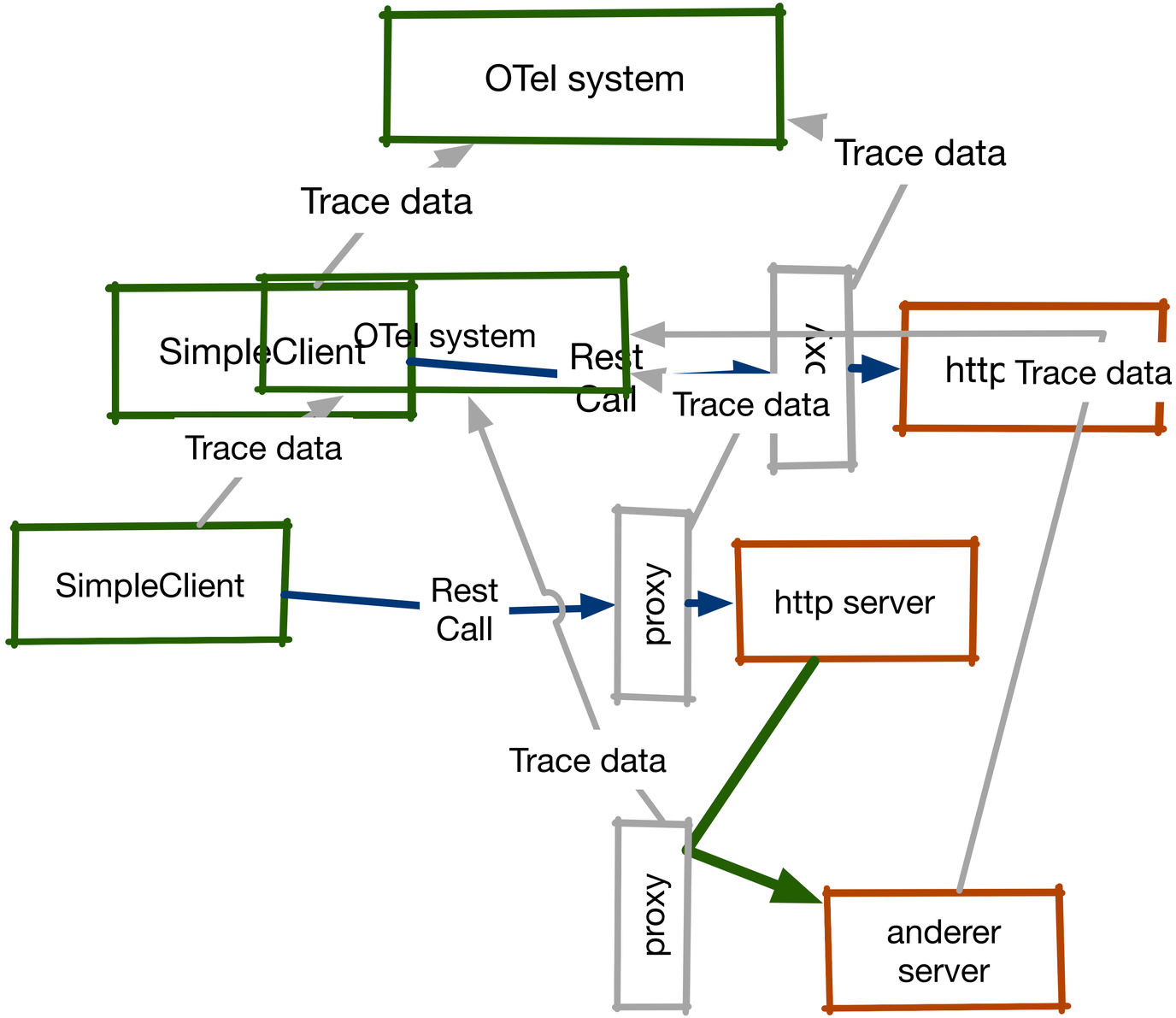
- Code wird zur Laufzeit angepasst
  - Python: Monkey-Patching
  - Java: Byte-Code Veränderung

```
uv run opentelemetry-instrument \
    --traces_exporter console,otlp \
    --metrics_exporter console \
    --service_name python-with-agent \
    --exporter_otlp_endpoint 0.0.0.0:4317 \
    python -m http.server
```

Proxy/Sidecar

Ähnlich aber anders

- Proxy empfängt Traffic und fügt header ein
  - zB Envoy bei Istio
  - Prozess selbst muss trotzdem Kontext weitergeben



Red Hat

OBI

Nein, kein Baumarkt

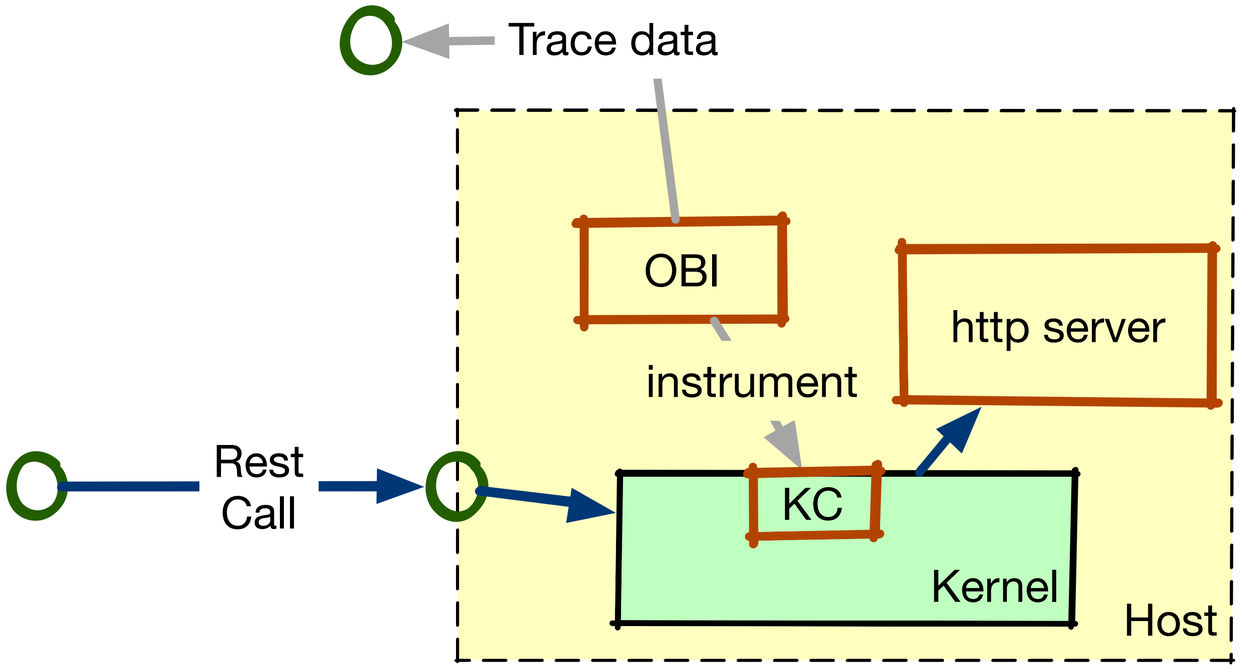
- **O**penTelemetry e**B**pf **I**nstrumentation
  - Gestartet als ‚Grafana Beyla‘
- eBPF = enhanced Berkeley Packet Filter
  - „Plugins für den Kernel“
  - Nur für Linux (und wohl *BSD)

Siehe auch https://www.pilhuhn.de/docs/2026/tracing_obi/

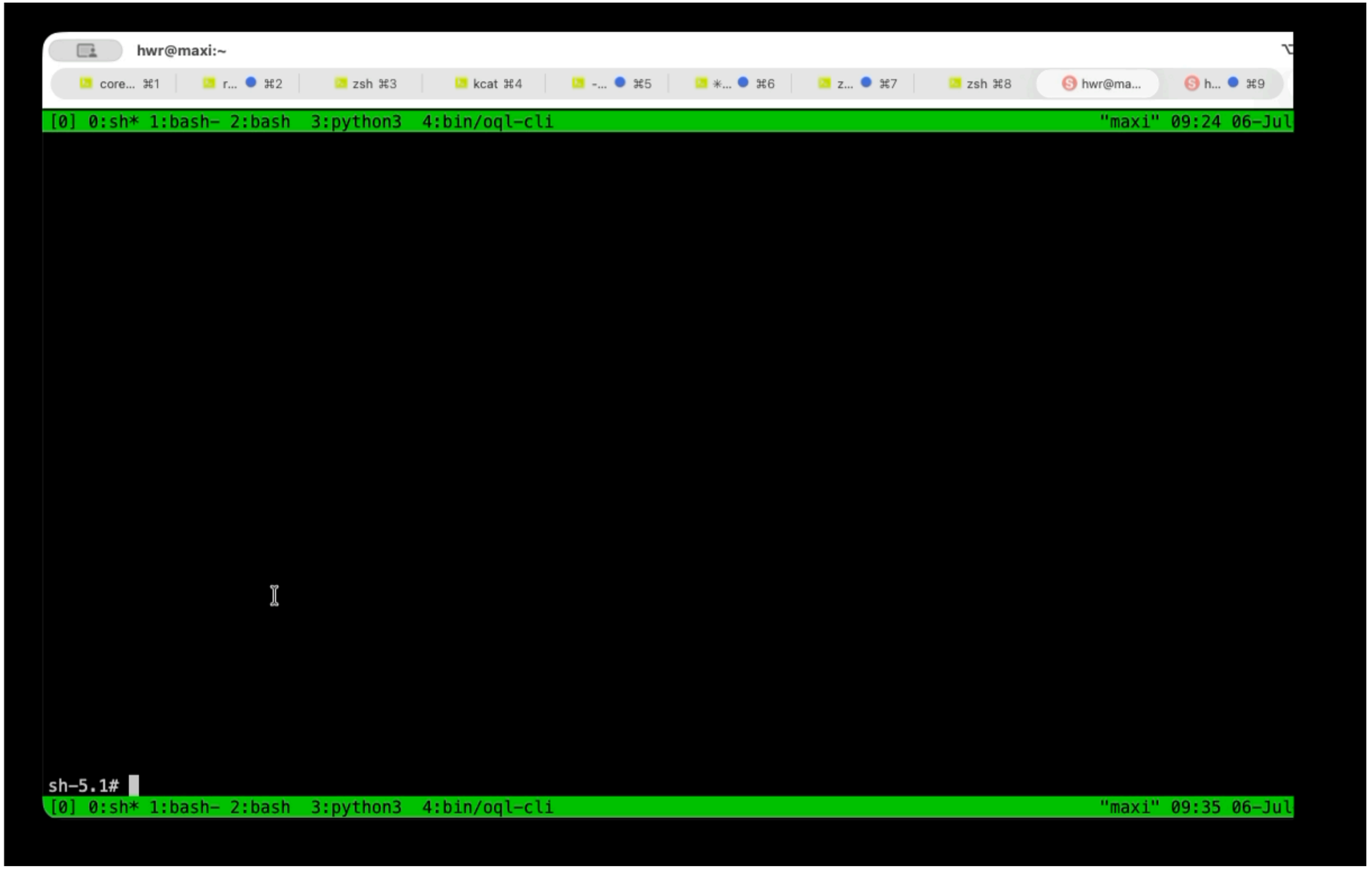
Red Hat

OBI (2)

- Probe „*KC*“ wird in den Kernel geladen
- Probe sieht den Traffic, liest http-Header
- OBI (Userspace) sendet trace data
- Probe kann auch Header modifizieren



OBI (3)


hwr@maxi:~
core... ⌘1
r... ⌘2
zsh ⌘3
kcat ⌘4
-... ⌘5
*... ⌘6
z... ⌘7
zsh ⌘8
hwr@ma...
h... ⌘9
[0] 0:sh* 1:bash- 2:bash 3:python3 4:bin/oql-cli
"maxi" 09:24 06-Jul
sh-5.1#
[0] 0:sh* 1:bash- 2:bash 3:python3 4:bin/oql-cli
"maxi" 09:35 06-Jul

Red Hat

Fazit

Viele Wege führen nach Rom

- Auto-Instrumentierung
  - Quickstart
  - Masse
- Manuelle Instrumentierung + Bibliotheken
  - Anwendung auf Geschäftslogik

Red Hat